Chapter 1
A simple controller for line following of sailboats

Luc Jaulin and Fabrice Le Bars

Abstract This paper proposes a simple controller for sailboat robots. The resulting controller is simple to implement and its parameters are easy to tune. Its complexity is low enough to be applicable for sailing robots with very limited computation power. The presentation contains all the necessary details to allow a fast and reliable implemention of a sailbot robot controller which follows a line. The paper also presents a simple collision avoidance strategy based on interval analysis.

1.1 Introduction

This paper deals with the problem of controlling a sailboat robot. It describes in a pedagogical way a controller that has been made generic enough to be used for a large class of sailboat robots. Note that it is the controller that has been implemented on the sailboat robot Vaimos [1] and has been proved to be very efficient and robust in several convincing experiments. We did our best to make the controller understandable by students that are not specialists in sailboat robotics. Our problem is motivated by the microtransat challenge [2][20] where autonomous sailboat robots have to cross the Atlantic ocean from East to West. Figure 1.1 illustrates the control loop to be considered in this paper. Sailboats are nonlinear hybrid systems involving strong perturbations such as waves. Moreover, to our knowledge, no realistic state equations are available for sailboats. For these reasons, existing methods from control theory [14] may not be appropriate for building reliable controllers for sailboat robots. Now, sailboats have been designed for thousand of years to be easily controlled by humans. A pragmatic approach that mimics the control strategy of sailors is thus chosen here to build reliable controllers.

The paper is organized as follows. Section 2 describes the pragmatic controller. An extension taking into account that the Earth is not flat is then considered in Section 3. In Section 4, an elementary collision avoidance strategy is proposed. A conclusion is given in Section 5.

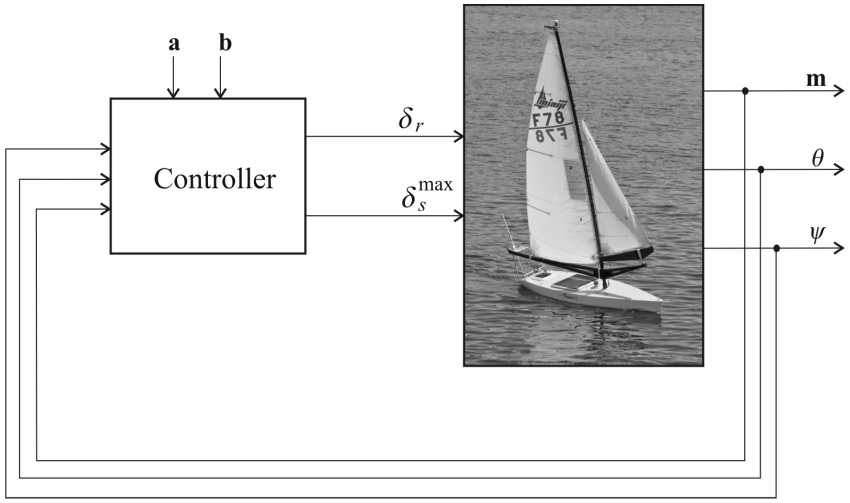


Fig. 1.1 Controller of a sailboat robot

1.2 Controller

A classical approach to build controllers is to take a realistic model of the system to be controlled (such as [6] for the sailboat) and then to use classical control methods to get the controller. Here, we follow a pragmatic approach influenced by the potential field strategy proposed by [18] for sailboat robots (see also [4]). The sailboat is assumed to have three sensors and two actuators. The controller will have some parameters which are easy to tune, some reference variables and one binary state variable. Let us now describe all of them.

Sensors. The heading θ of the robot is measured by a compass. The angle of the wind ψ is returned by a weather vane (even if this sensor can sometimes be omitted as shown in [22]). The position **m** is given by a GPS.

Actuators. The inputs of the robot are the angle of the rudder δ_r and the maximum angle for the sail $\delta_s^{\max}$ (which is directly related to the length of the mainsheet).

Parameters. $\delta_r^{\max}$ is the maximal angle of the rudder (we shall set $\delta_r^{\max} = \frac{\pi}{4}$), r is the cutoff distance (i.e., we want that the distance of the boat to the line be less than r; we shall choose $r = 50$m), γ_∞ is the incidence angle (we take $\gamma_\infty = \frac{\pi}{4}$) and ζ is the close hauled angle (we choose $\zeta = \frac{\pi}{3}$).

References. Two points $\mathbf{a}, \mathbf{b}$ which define the line to be followed.

State variable. This will be a discrete variable $q \in \{-1, 1\}$ corresponding to the favored tack.

We propose the following algorithm to describe the controller [12]. This algorithm (presented in [11] in a theoretical form) is given in its low level form to allow a fast and reliable implementation by anyone who wants to build a controller for sailboat robots.

Function in: $\mathbf{m}, \theta, \psi, \mathbf{a}, \mathbf{b}$; out: $\delta_r, \delta_s^{\max}$; inout: q

1 $e = \det\left(\frac{\mathbf{b}-\mathbf{a}}{\|\mathbf{b}-\mathbf{a}\|}, \mathbf{m}-\mathbf{a}\right)$
2 if $|e| > \frac{r}{2}$ then $q = \text{sign}(e)$
3 $\varphi = \text{atan2}(\mathbf{b}-\mathbf{a})$
4 $\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\text{atan}\left(\frac{e}{r}\right)$
5 if $\cos(\psi-\theta^*) + \cos\zeta < 0$
6 or $(|e| < r$ and $(\cos(\psi-\varphi) + \cos\zeta < 0))$
7 then $\bar{\theta} = \pi + \psi - q.\zeta$.
8 else $\bar{\theta} = \theta^*$
9 end
10 if $\cos\left(\theta - \bar{\theta}\right) \geq 0$ then $\delta_r = \delta_r^{\max}.\sin\left(\theta - \bar{\theta}\right)$
11 else $\delta_r = \delta_r^{\max}.\text{sign}\left(\sin\left(\theta - \bar{\theta}\right)\right)$
12 $\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi-\bar{\theta}\right)+1}{2}\right)$.

The controller has one state variable $q \in \{-1, 1\}$. This is why it is both an input and an output variable of the algorithm.

Step 1. We compute the algebraic distance of the boat to the line. If $e > 0$ the robot is on the left of the line and if $e < 0$, it is on the right. In practice, since the Earth is not flat, it is important to have a reasonable distance between **a** and **b** (less than 100km). In the formula, the determinant between two vectors is defined by

$$\det(\mathbf{u}, \mathbf{v}) = u_1 v_2 - v_1 u_2.$$

Step 2. If $|e| > \frac{r}{2} = 25m$, we are quite far from the line and the tack variable q is allowed to change its value. If for instance $e > 25m$, then q will be set to 1 and will keep this value until $e < -25m$.

Step 3. We compute the angle φ of the line to be followed (see Figure 1.2). In the statement, $\text{atan2}(\mathbf{u}) = \text{atan2}(u_1, u_2)$ represents the angle of the two-dimensional vector **u** with respect to East.

Step 4. We compute the nominal angle θ^* (see Figure 1.2) given by

$$\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.atan\left(\frac{e}{r}\right).$$

This expression for θ^* makes the line attractive. When $e = \pm\infty$, we have $\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\left(\pm\frac{\pi}{2}\right) = \varphi \pm \gamma_\infty$, i.e., the robot has a heading which corresponds to the angle γ_∞. For the cutoff distance $e = \pm r$, we have $\theta^* = \varphi \pm \frac{2.\gamma_\infty}{\pi}.\frac{\pi}{4} = \varphi \pm \frac{\gamma_\infty}{2}$ and for $e = 0$, $\theta^* = \varphi$, which corresponds to the direction of the line. As illustrated by Figure 1.3, some directions θ^* may be inconsistent with the current wind.

Step 5. If $\cos(\psi-\theta^*) + \cos\zeta < 0$, the course θ^* corresponds to a direction which is too close to the wind which the boat is unable to follow (see Figure 1.4). The course θ^* is thus inconsistent with the wind. If this happens, we choose a close hauled mode, i.e., the new feasible direction becomes $\bar{\theta} = \pi + \psi \pm \zeta$ (at Step 7). Figure 1.5 represents the corresponding vector field. Thin arrows correspond to the

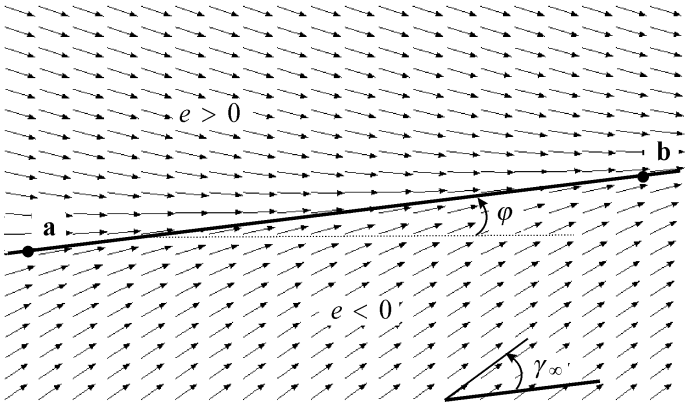


Fig. 1.2 Nominal vector field θ^* that the robot has follow when possible. The variable γ_∞ corresponds to the incidence angle when the distance to the boat is large ($|e| > 500m$).

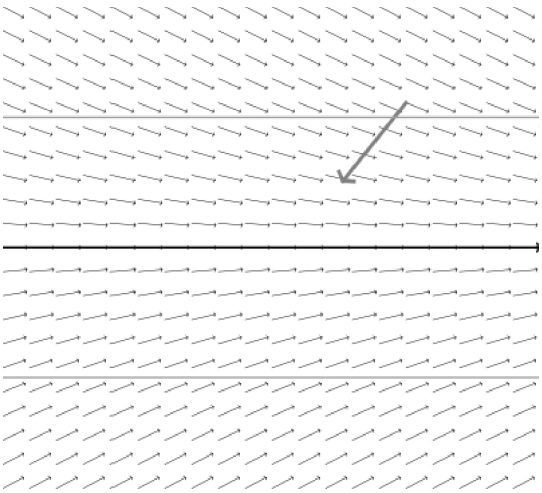
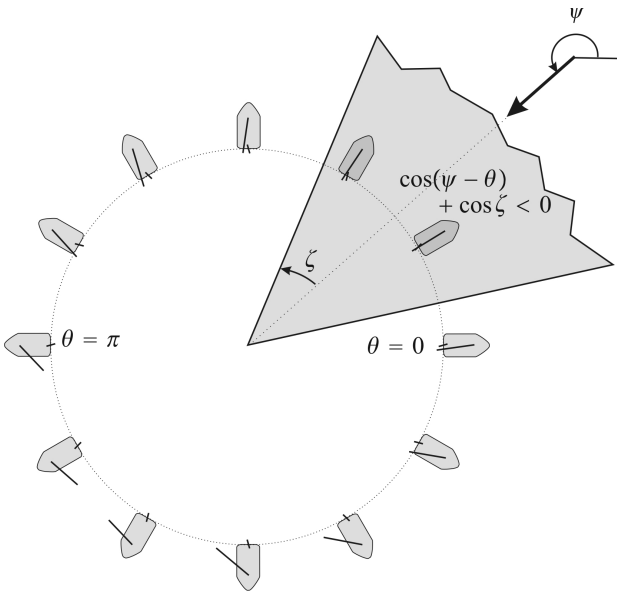


Fig. 1.3 The normal field may be inconsistent with the wind (bold arrow); The line to be followed corresponds to the x-axis.

nominal field and thick arrows correspond to the corrected field (when the latter is different from the nominal field) In this representation, we have removed the hysteresis effect induced by the tack variable q (it is equivalent to saying that we always have $q = \text{sign}(e)$).

Step 6. This step implements what we call the *keep close hauled strategy*. If $|e| < \underline{r}$ and $\cos(\psi - \varphi) + \cos\zeta < 0$, we force the close hauled mode even if the route $\bar{\theta}$ is feasible. For efficiency reasons, when the line is against the wind, we



.

Fig. 1.4 Some directions for the sailboat are not feasible. These unfeasible courses forms the no-go zone painted grey.

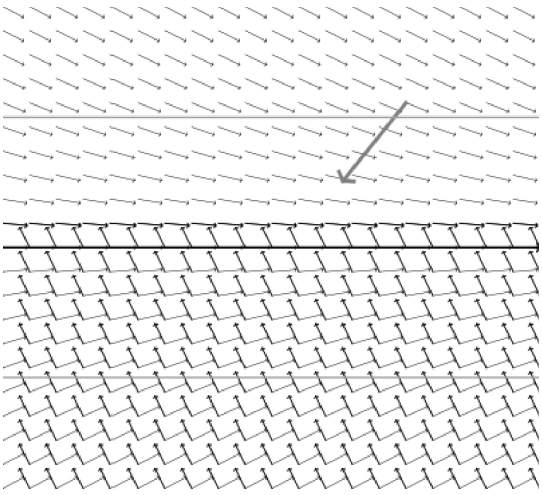


Fig. 1.5 Vector field provided by the algorithm if we remove Step 6. Thin arrows correspond to nominal routes. Thick arrows correspond to corrected routes when the nominal route is not feasible.

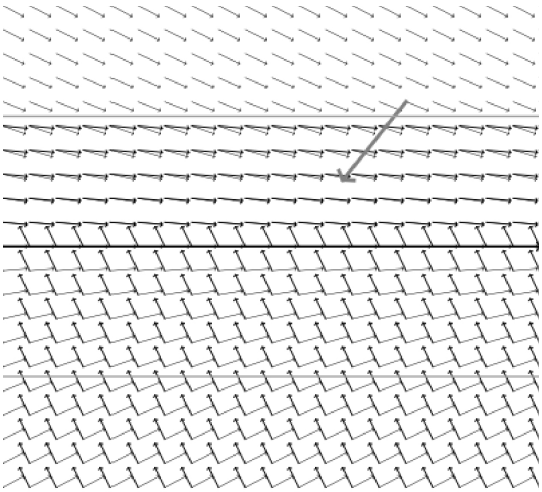


Fig. 1.6 Vector field provided by the algorithm including Step 6. Thin arrows correspond to nominal routes. Thick arrows correspond to corrected routes based on the keep close hauled strategy.

do not want to loose against the wind. This is illustrated by Figure 1.6 where the conventions are those of Figure 1.5. Note that in this figure we took $\zeta = \frac{\pi}{3}$ (which corresponds to a boat that has difficulties in going upwind in a close hauled mode) which makes the line against the wind. A video with more explanations can be found at [9].

Step 7. We are in the close-hauled mode and we choose. $\bar{\theta} = \pi + \psi - q.\zeta$ (the wind direction plus or minus the close hauled angle ζ). The hysteresis variable q makes it possible to keep the current tack for 25 more meters to the line even if the line to be followed has been crossed.

Step 8. If the nominal route is satisfactory, we keep it.

Step 10. At this level, the feasible course θ has been chosen and we want to tune the rudder. If the robot has a consistent direction, we perform a proportional control with respect to the error $\sin\left(\theta - \bar{\theta}\right)$. This is illustrated by Figure 1.7, Quarters 1 and 2.

Step 11. If the robot does not have a consistent direction, i.e. $\cos\left(\theta - \bar{\theta}\right) < 0$, the rudder is tuned at the maximum (see Figure 1.7, Quarters 3 and 4).

Step 12. The length of the mainsheet is tuned with respect to the cardioid relation $\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi-\bar{\theta}\right)+1}{2}\right)$ [12]. Note that if $\psi - \bar{\theta} = \pm\pi$ (the boat is wind ahead), $\delta_s^{\max} = 0$ whereas $\delta_s^{\max} = \frac{\pi}{2}$ if the wind comes from abeam.

Remark. In practice, in a direct mode (i.e. when φ corresponds to a feasible course), a bias of 10 meters could occur, i.e. the distance to the line does not converge to 0. An integrator term could avoid this bias. To implement the integrator, it

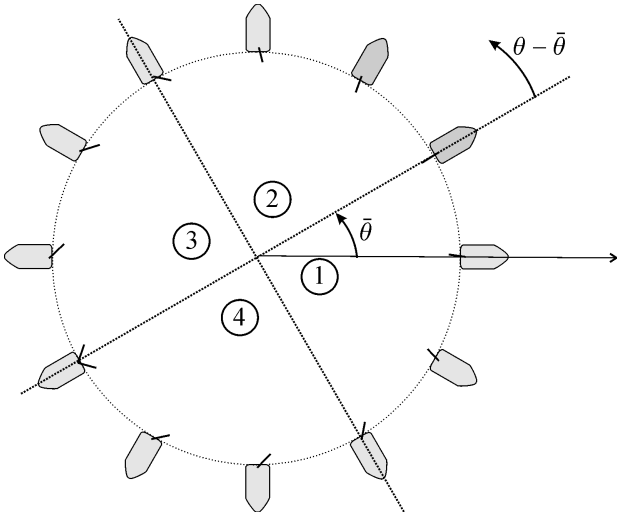


Fig. 1.7 Four quarter technique for the tuning of the rudder to go in the route $\bar{\theta}$. Quarters 1 and 2, we have $\cos\left(\theta - \bar{\theta}\right) \geq 0$ and a proportional control is applied; Quarters 3 and 4, we have $\cos\left(\theta - \bar{\theta}\right) \leq 0$ and a bang-bang control is chosen; Quarters 2 and 3, we have $\sin\left(\theta - \bar{\theta}\right) \geq 0$ then we turn left ($\delta_r > 0$); Quarters 1 and 4, we have $\sin\left(\theta - \bar{\theta}\right) \leq 0$ then we turn right ($\delta_r < 0$).

suffices to replace Step 4 by the two following statements:

$$\begin{cases} z = z + \alpha * dt * e \\ \theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\text{atan}\left(\frac{e+z}{r}\right) \end{cases}$$

where dt is the sampling time. The variable z corresponds to the value of the integrator and converges to the bias we had without the integrator. The coefficient α should be small enough to avoid any change in the behavior of the controlled sailboat. For instance, if for $e = 10m$ for 100 sec. we want a correction of 1m, we shall take $\alpha = 0.001$. As soon as the distance to the line is higher than 50 meters (during the initialization, for instance), if the the robot switches to another line (as it is the case when a line is validated), or if the robot switches to a close hauled mode, the integrator z should be forced to 0.

1.3 Earth is not flat

We now want to take into account the fact that the Earth is not flat. We shall now adapt the controller of the previous section to our new situation. Denote by ℓ_x and ℓ_y the longitude and the latitude of a point which is located at the surface of the Earth. The transformation into the geographic coordinate system is given by

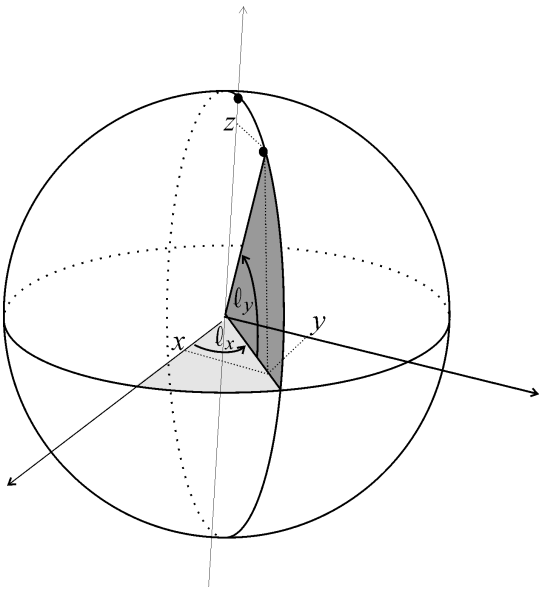


Fig. 1.8 Geographic reference frame

$$\mathscr{T} : \begin{pmatrix} \ell_x \\ \ell_y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \rho \cos \ell_y \cos \ell_x \\ \rho \cos \ell_y \sin \ell_x \\ \rho \sin \ell_y \end{pmatrix} \tag{1.1}$$

where $\rho = 6371000m$ is the radius of the Earth (see Figure 1.8).

Consider three points $\mathbf{a}, \mathbf{b}, \mathbf{m}$ at the surface of the Earth (see Figure 1.9). The vector

$$\mathbf{n} = \frac{\mathbf{a} \wedge \mathbf{b}}{\|\mathbf{a}\| \, \|\mathbf{b}\|}$$

is normal to the plane $(\mathbf{oab})$ and has a norm equal to 1. The algebraic distance from $\mathbf{m}$ to the plane $(\mathbf{oab})$ is given by

$$e = \mathbf{m}^{\mathrm{T}}.\mathbf{n}.$$

Let us differentiate the relation (1.1) We get

$$\begin{pmatrix} dx \\ dy \\ dz \end{pmatrix} = \mathbf{J}. \begin{pmatrix} d\ell_x \\ d\ell_y \\ d\rho \end{pmatrix}$$

where

$$\mathbf{J} = \begin{pmatrix} -\rho \cos \ell_y \sin \ell_x & -\rho \sin \ell_y \cos \ell_x & \cos \ell_y \cos \ell_x \\ \rho \cos \ell_y \cos \ell_x & -\rho \sin \ell_y \sin \ell_x & \cos \ell_y \sin \ell_x \\ 0 & \rho \cos \ell_y & \sin \ell_y \end{pmatrix}.$$

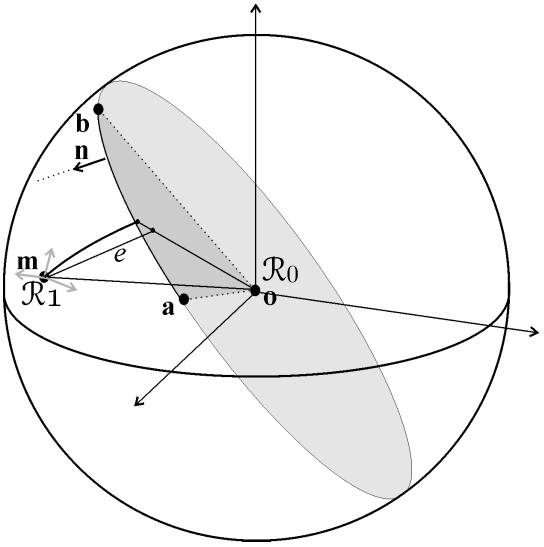


Fig. 1.9 Line (**ab**) to be followed

This formula can be used to find the the geographic coordinates of the cardinal directions. For instance, the vector corresponding to the East is given by the first column. Equivalently, we are able to build a East-North-Elevation frame $\mathscr{R}_1$ around the robot (in grey on Figure 1.9). The corresponding rotation matrix is obtained by normalizing each column of the Jacobian matrix **J**:

$$\mathbf{R} = \begin{pmatrix} -\sin\ell_x & -\sin\ell_y\cos\ell_x & \cos\ell_y\cos\ell_x \\ \cos\ell_x & -\sin\ell_y\sin\ell_x & \cos\ell_y\sin\ell_x \\ 0 & \cos\ell_y & \sin\ell_y \end{pmatrix}.$$

The transformation relation to move from the geographic frame $\mathscr{R}_0$ to the robot frame $\mathscr{R}_1$ is

$$\mathbf{v}_{|\mathscr{R}_1} = \mathbf{R}^{\mathrm{T}}.\mathbf{v}_{|\mathscr{R}_0}. \tag{1.2}$$

To get φ, take the vector $\mathbf{b}-\mathbf{a}$, express it in the $\mathscr{R}_1$ (using (1.2)) frame, project it into the $(\mathbf{i},\mathbf{j})$ frame (by selecting the two first rows) and take its argument (using the *atan2* function). This gives

$$\varphi = \mathrm{atan2}\left(\mathbf{M}.(\mathbf{b}-\mathbf{a})_{|\mathscr{R}_0}\right),$$

where

$$\mathbf{M} = \begin{pmatrix} -\sin\ell_x^{\mathbf{m}} & \cos\ell_x^{\mathbf{m}} & 0 \\ -\cos\ell_x^{\mathbf{m}}\sin\ell_y^{\mathbf{m}} & -\sin\ell_x^{\mathbf{m}}\sin\ell_y^{\mathbf{m}} & \cos\ell_y^{\mathbf{m}} \end{pmatrix}.$$

The resulting controller is given by the following table

Function in : $\theta, \psi, \ell_x^{\mathbf{a}}, \ell_y^{\mathbf{a}}, \ell_x^{\mathbf{b}}, \ell_y^{\mathbf{b}}, \ell_x^{\mathbf{m}}, \ell_y^{\mathbf{m}}$;
out: $\delta_r, \delta_s^{\max}$; inout: q

$\mathbf{a} = \mathscr{T}\left(\ell_x^{\mathbf{a}}, \ell_y^{\mathbf{a}}\right)$; $\mathbf{b} = \mathscr{T}\left(\ell_x^{\mathbf{b}}, \ell_y^{\mathbf{b}}\right)$; $\mathbf{m} = \mathscr{T}\left(\ell_x^{\mathbf{m}}, \ell_y^{\mathbf{m}}\right)$;
$e = \mathbf{m}^{\mathrm{T}}.\frac{\mathbf{a}\wedge\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$;
if $|e| > \frac{r}{2}$ then $q = \mathrm{sign}(e)$;
$\mathbf{M} = \begin{pmatrix} -\sin\ell_x^{\mathbf{m}} & \cos\ell_x^{\mathbf{m}} & 0 \\ -\cos\ell_x^{\mathbf{m}}\sin\ell_y^{\mathbf{m}} & -\sin\ell_x^{\mathbf{m}}\sin\ell_y^{\mathbf{m}} & \cos\ell_y^{\mathbf{m}} \end{pmatrix}$;
$\varphi = \mathrm{atan2}\left(\mathbf{M}.(\mathbf{b}-\mathbf{a})\right)$;
$\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\mathrm{atan}\left(\frac{e}{r}\right)$;
if $\cos(\psi-\theta^*)+\cos\zeta < 0$
  or $(|e| < r$ and $(\cos(\psi-\varphi)+\cos\zeta < 0))$
    then $\bar{\theta} = \pi + \psi - q.\zeta$;
    else $\bar{\theta} = \theta^*$;
end;
if $\cos\left(\theta-\bar{\theta}\right) \geq 0$ then $\delta_r = \delta_r^{\max}.\sin\left(\theta-\bar{\theta}\right)$;
else $\delta_r = \delta_r^{\max}.\mathrm{sign}\left(\sin\left(\theta-\bar{\theta}\right)\right)$;
$\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi-\bar{\theta}\right)+1}{2}\right)$.

1.4 Avoiding collisions

In this section, we assume again (as in Section 1.2) that the Earth can be approximated by a plane on which a Cartesian frame *Oxy* is available. Consider the situation where m other boats are detected at time $t = 0$ by our robot, for instance using an AIS (Automatic Identification System). Note that the time $t = 0$ is chosen as the reference time and does not correspond to the beginning of the mission. We assume that we measure the speed and the position of these boats with a known accuracy. The speed is considered as constant. More precisely, the position of all these boats is assumed to satisfy

$$\mathbf{m}^i(t) = \mathbf{a}^i.t + \mathbf{b}^i, \;\; i \in \{1, \ldots m\}$$

where $\mathbf{a}^i$ and $\mathbf{b}^i$ are vectors of $\mathbb{R}^2$ which correspond to the speed vector and the initial position of each boat. Since we measure these two quantities with a known accuracy, we have two boxes $[\mathbf{a}^i]$ and $[\mathbf{b}^i]$ which enclose $\mathbf{a}^i$ and $\mathbf{b}^i$, respectively. Moreover, we assume that the trajectory of our robot is described by

$$\mathbf{m}^0(t) = \mathbf{a}^0.t + \mathbf{b}^0.$$

The two vectors $\mathbf{a}^0$ and $\mathbf{b}^0$ can be obtained by taking into account the characteristics of the line to be followed (see Section 1.2) and the speed of our robot (which can be estimated from the previous GPS measurements). Again, we assume that we have two boxes $[\mathbf{a}^0]$ and $[\mathbf{b}^0]$ which enclose $\mathbf{a}^0$ and $\mathbf{b}^0$. We shall propose a method based on interval analysis [17] [13] [21] to prove that our robot's trajectory is safe.

Interval analysis is a numerical tool able to deal with nonlinear problems involving uncertainties (see, e.g. [5], [15], [10], [16] and [19] in the context of robotics). It has also been used in the context of sailboat robotics [8]. First, recall some basic interval operations that will be used later.

$$\begin{array}{l}[x^-,x^+]+[y^-,y^+]=[x^-+y^-,x^++y^+]\\ [x^-,x^+]-[y^-,y^+]=[x^--y^+,x^+-y^-]\\ [x^-,x^+]*[y^-,y^+]=[\min(x^-y^-,x^+y^-,x^-y^+,x^+y^+),\max(x^-y^-,x^+y^-,x^-y^+,x^+y^+)].\end{array}$$

For instance

$$[2,3]*[-1,2]+[3,4]=[-3,6]+[3,4]=[0,10].$$

Proposition 1. If for all $i\in\{1,\dots m\}$,

$$\left\{\begin{array}{cc} 0\notin\left(\left[a_x^0\right]-\left[a_x^i\right]\right)*[0,t_{\max}]+\left[b_x^i\right]-\left[b_x^0\right] & \text{or}\\ 0\notin\left(\left[a_y^0\right]-\left[a_y^i\right]\right)*[0,t_{\max}]+\left[b_y^i\right]-\left[b_y^0\right] & \text{or}\\ 0\notin\left(\left[a_y^0\right]-\left[a_y^i\right]\right)*\left(\left[b_x^i\right]-\left[b_x^0\right]\right)-\left(\left[a_x^0\right]-\left[a_x^i\right]\right)*\left(\left[b_y^i\right]-\left[b_y^0\right]\right) & \end{array}\right.$$

then the trajectory of the robot is collision free inside the time interval $[0,t_{\max}]$.

Proof. Our trajectory is *collision-free* inside an interval $[0,t_{\max}]$ if

$$\forall i\in\{1,\dots m\},\forall t\in[0,t_{\max}],\mathbf{m}^i(t)\neq\mathbf{m}^0(t).$$

By taking the contrapositive of this proposition, we get that if the system

$$\left\{\begin{array}{c}\left(\mathbf{a}^0-\mathbf{a}^i\right).t+\mathbf{b}^0-\mathbf{b}^i=0,\\ t\in[0,t_{\max}]\\ \mathbf{a}^0\in\left[\mathbf{a}^0\right],\mathbf{b}^0\in\left[\mathbf{b}^0\right],\mathbf{a}^i\in\left[\mathbf{a}^i\right],\mathbf{b}^i\in\left[\mathbf{b}^i\right],\end{array}\right.$$

or equivalently

$$\left\{\begin{array}{c}\left(a_x^0-a_x^i\right)t+b_x^i-b_x^0=0\\ \left(a_y^0-a_y^i\right)t+b_y^i-b_y^0=0\\ t\in[0,t_{\max}]\\ a_x^0\in\left[a_x^0\right],b_x^0\in\left[b_x^0\right],a_y^0\in\left[a_y^0\right],b_y^0\in\left[b_y^0\right]\\ a_x^i\in\left[a_x^i\right],b_x^i\in\left[b_x^i\right],a_y^i\in\left[a_y^i\right],b_y^i\in\left[b_y^i\right]\end{array}\right.\tag{1.3}$$

has no solution for all $i\in\{1,\dots m\}$, then our trajectory is collision free. Now, from the two first lines of (1.3), we get

$$\left(a_y^0-a_y^i\right)\left(b_x^i-b_x^0\right)-\left(a_x^0-a_x^i\right)\left(b_y^i-b_y^0\right)=0.$$

The fundamental theorem of interval analysis [17] applied on the three equations terminates the proof. ■

Method for avoiding collisions. We propose to take $t_{\max}=10\,\text{min}$ and apply the following procedure every minute:

Step 1. *Normal mode*. Using proposition 1, check if the current course (with angle φ) that is followed by the robot is collision free. If it is not the case, go to Step 2.

Step 2. *Anchor mode*. Anchor (virtually) the robot for 10 minutes. Go to Step 1.

1.5 Conclusion

In this paper we have presented a simple controller to allow a sailboat robot to follow a line. The controller is easy to understand, implement, test and debug, compared to other more sophisticated controllers such as the one developed by Guillou [7] or by Bruder et. al. [3]. All computations can be performed using any cheap and low-powered microcontrollers, which is a key point in the context of sailboat robotics where the energy is highly limited. The controller can easily be adapted to build controllers which are less generic, but more efficient since they can be tuned on a particular sailboat. A simple collision avoidance strategy based on interval analysis has also been presented in the last part of the paper.

Acknowledgement

The robot *Vaimos* is the result of a collaboration between LPO (Laboratoire de Physique des Océans), RDT (Recherches et Développements Technologiques) of IFREMER (Institut Français de Recherche pour l'Exploitation de la Mer) and ENSTA-Bretagne (Ecole Nationale Supérieure de Techniques Avancées). The authors render thanks to LPO and RDT for having made available *Vaimos* to test the controllers presented in this paper. They also thank all people involved in the project: Y. Auffret, S. Barbot, L. Dussud, B. Forest, E Menut, S. Prigent, L. Quemeneur, P. Rousseaux (RDT, IFREMER); F. Gaillard, T. Gorgues, O. Ménage, J. Moranges, T. Terre (LPO) and B. Clément, Y. Gallou, O. Reynet, J. Sliwka and B. Zerr (ENSTA-Bretagne).

References

1. F. Le Bars and L. Jaulin. An experimental validation of a robust controller with the VAIMOS autonomous sailboat. In *International Robotic Sailing Conference*, Cardiff, Wales, England, 2012.
2. Y. Brière. *The first microtransat challenge* , `http://web.ensica.fr/microtransat`. ENSICA, 2006.
3. R. Bruder, B. Stender, and A. Schlaefer. Model sailboats as a testbed for artificial intelligence methods. In *International Robotic Sailing Conference*, Matosinhos, Portugal, 2009.
4. N.A. Cruz and J.C. Alves. Ocean sampling and surveillance using autonomous sailboats. In *International Robotic Sailing Conference*, Austria, 2008.
5. V. Drevelle and P. Bonnifait. High integrity gnss location zone characterization using interval analysis. In *ION GNSS*, 2009.
6. T.J. Gale and J.T. Walls. Development of a sailing dinghy simulator. *Simulation*, 74(3):167–179, 2000.

7. G. Guillou. *Architecture multi-agents pour le pilotage automatique des voiliers de compétition et extensions algébriques des réseaux de Petri*. PhD dissertation, Université de Bretagne, Brest, France, 2011.
8. P. Herrero, L. Jaulin, J. Vehi, and M. A. Sainz. Guaranteed set-point computation with application to the control of a sailboat. *International Journal of Control Automation and Systems*, 8(1):1–7, 2010.
9. http://youtu.be/pHteidmZpnY.
10. L. Jaulin. A Nonlinear Set-membership Approach for the Localization and Map Building of an Underwater Robot using Interval Constraint Propagation. *IEEE Transaction on Robotics*, 25(1):88–98, 2009.
11. L. Jaulin and F. Le Bars. An interval approach for stability analysis; Application to sailboat robotics. *submitted to IEEE Transaction on Robotics*, 2012.
12. L. Jaulin, F. Le Bars, B. Clément, Y. Gallou, O. Ménage, O. Reynet, J. Sliwka, and B. Zerr. Suivi de route pour un robot voilier. In *CIFA 2012*, Grenoble (France), 2012.
13. R. B. Kearfott and V. Kreinovich, editors. *Applications of Interval Computations*. Kluwer, Dordrecht, the Netherlands, 1996.
14. H.K. Khalil. *Nonlinear Systems, Third Edition*. Prentice Hall, 2002.
15. F. LeBars, J. Sliwka, O. Reynet, and L. Jaulin. State estimation with fleeting data. *Automatica*, 48(2):381–387, 2012.
16. P. Lucidarme, L. Hardouin, and J. L. Paillat. Variable geometry tracked vehicle (vgtv) prototype : conception, capability and problems. In *Humans Operating Unmanned Systems (HUMOUS) conference*, pages 115–126, France - Brest, 2008.
17. R. E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1966.
18. C. Petres, M. Romero Ramirez, and F. Plumet. Reactive path planning for autonomous sailboat. In *IEEE International Conference on Advanced Robotics*, pages 1–6, 2011.
19. N. Ramdani and P.Poignet. Robust dynamic experimental identification of robots with set membership uncertainty. *IEEE/ASME Transactions on Mechatronics*, 10(2):253–256, 2005.
20. C. Sauze and M. Neal. An autonomous sailing robot for ocean observation. In *proceedings of TAROS 2006*, pages 190–197, Guildford, UK, 2006.
21. W. Tucker. The Lorenz attractor exists. *Comptes Rendus de l'académie des Sciences*, 328(12):1197–1202, 1999.
22. K. Xiao, J. Sliwka, and L. Jaulin. A wind-independent control strategy for autonomous sailboats based on voronoi diagram. In *CLAWAR 2011 (best paper award)*, Paris, 2011.